

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA: CNTT



NHẬP MÔN MÃ HÓA - MẬT MÃ
BÁO CÁO ĐỒ ÁN

Giảng viên hướng dẫn: Nguyễn Đình Thúc

Sinh viên:

1. Nguyễn Đức Thắng - 19120654
2. Phạm Thị Nguyệt - 19120607
3. Phạm Văn Thành - 19120659
4. Trương Công Thành - 19120660
5. Nguyễn Văn Thịnh - 19120667

Thành phố Hồ Chí Minh, 10 tháng 01 năm 2022

MỤC LỤC

1. Phân công công việc.....	1
2. Nội dung.....	1
2.1 Lý thuyết.....	1
2.2 Vận dụng.....	6
2.2.1. Mô tả tổng quan các dịch vụ:.....	6
2.2.2. Các thành phần của source code:.....	8
2.2.3. Cách biên dịch và chạy chương trình:.....	8
2.2.4. Mã giả các thuật toán: (RSA, AES).....	8
2.2.5. Hướng dẫn sử dụng chương trình:.....	9
2.3 Mở rộng.....	12
2.3.1. Ưu và nhược điểm của hệ thống:.....	12
2.3.2. Công nghệ thế giới đang sử dụng.....	12
2.3.3. Hướng cải tiến.....	14
3. Demo + source.....	16
4. Tài liệu tham khảo.....	17

1. Phân công công việc

STT	MSSV	Họ tên	Mức độ đóng góp
1	19120654	Nguyễn Đức Thắng	20%
2	19120607	Phạm Thị Nguyệt	20%
3	19120659	Phạm Văn Thành	20%
4	19120660	Trương Công Thành	20%
5	19120667	Nguyễn Văn Thịnh	20%

2. Nội dung

2.1 Lý thuyết

2.1.1 RSA Algorithm:

- *Các hàm chức năng:*

power(a, d, n): tính giá trị của a mũ d modulo cho n

MillerRabin(N,m): Kiểm tra tính nguyên tố của số tự nhiên N

PrimeTest(N,K): Hay FermatTest, kiểm tra số tự nhiên N

generate_prime_candidate(length): sinh số ngẫu nhiên length bits

generatePrimeNumber(length): sinh số nguyên tố ngẫu nhiên length bits

GCD(a,b): tìm UCLN của 2 số a, b

Bezout(E, eulerTotient): cài đặt định lý bezout về số dư của phép chia đa thức

encryption(my_img): hàm cài đặt xử lý mã hóa ảnh

show_EncryptImage(enc): hiển thị ảnh sau khi đã mã hóa

decryption(enc): hàm cài đặt xử lý giải mã ảnh

Gcd(a, b): Tìm UCLN của 2 số a, b

- ***Giải tích thuật toán***

B1: Tạo 2 số nguyên số lớn ngẫu nhiên p, q

$p = \text{random_prime}(\text{halfkeylength})$

$q = \text{random_prime}(\text{halfkeylength})$

B2: Tính $n = pq$, hàm số euler $\phi(n) = (p-1) \times (q-1)$

B3: Tìm e là khóa công khai và là một số nguyên tố lớn ngẫu nhiên khác, sao cho $\text{gcd}(e, \phi) = 1$

B4: Tìm $d = \text{bezout}(e, \phi)$ d là khóa bí mật

2.1.2 AES Algorithm

- ***Các hàm chức năng***

Open-source

- ***Giải thích thuật toán***

Lưu ý:

+ AES là thuật toán mã hóa khối, mỗi khối 128 bits

+ Khóa (key) của AES có thể là 128/192/256 bits

Thực thi thuật toán:

AES chính xác là xử lý trên bytes hơn là trên bit. Do đó, khi xử lý trên khối 128 bits, có nghĩa là thuật toán xử lý 16 bytes dữ liệu đầu vào trong một lần.

Số lần thực thi/vòng (round) dựa trên độ dài khóa:

+ 128 bit key – 10 rounds

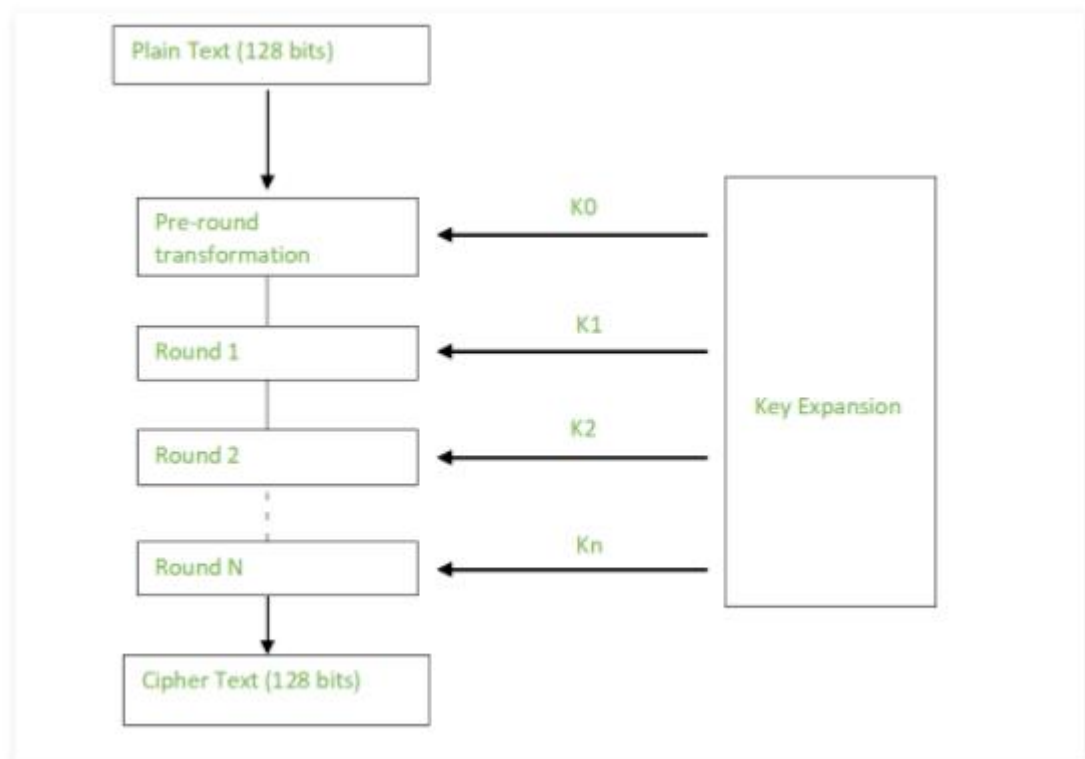
+ 192 bit key – 12 rounds

+ 256 bit key – 14 rounds

Các bước thực thi trong một round

- Tạo round keys

Key ban đầu (initial key) sẽ được tạo và sử dụng để tạo(mở rộng) ra key ở trong round mã hóa tiếp theo. Key được tạo dựa trên thuật toán Key Schedule



- Một round mã hóa Encryption:

AES làm việc với khối dữ liệu 128 bits (16 bytes), nên nó biến dữ liệu thành một ma trận 4x4, mỗi phần tử là 1 byte như hình:

```
[ b0 | b4 | b8 | b12 |  
| b1 | b5 | b9 | b13 |  
| b2 | b6 | b10 | b14 |  
| b3 | b7 | b11 | b15 ]
```

SubBytes sẽ làm công việc thay thế (substitution), còn ShiftRows và MixColumn sẽ thực hiện hoán vị (permutation)

- SubBytes

Mỗi byte sẽ được thay thế bằng một byte khác dựa trên việc tra cứu bảng có tên là S-box. Các byte thay thế sẽ không giống với byte trước đó.

Kết thúc SubBytes, ta sẽ có một ma trận 4x4 như ban đầu nhưng giá trị phần tử của nó đã được thay đổi.

VD S-box:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

- ShiftRows

Đây là được đầu của việc hoán vị. Mỗi hàng sẽ được ‘xê dịch’ với một số lần cụ thể:

+ Hàng 1: không dịch chuyển

+ Hàng 2: 1 lần sang bên trái

+ Hàng 3: 2 lần sang bên trái

+ Hàng 4: 3 lần sang bên trái

$$\begin{bmatrix} b_0 & b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 & b_7 \\ b_8 & b_9 & b_{10} & b_{11} \\ b_{12} & b_{13} & b_{14} & b_{15} \end{bmatrix} \rightarrow \begin{bmatrix} b_0 & b_1 & b_2 & b_3 \\ b_5 & b_6 & b_7 & b_4 \\ b_{10} & b_{11} & b_8 & b_9 \\ b_{15} & b_{12} & b_{13} & b_{14} \end{bmatrix}$$

- MixColumns

Bản chất của pha là một phép nhân ma trận. Mỗi cột sẽ được nhân với một ma trận cụ thể vị trí các byte sẽ được thay đổi

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Lưu ý: Ở round cuối sẽ không diễn ra pha MixColumns.

- Add round keys:

Kết quả của MixColumns sẽ được đem đi XOR với round key của round hiện tại. Lúc này, 16 bytes sẽ trở thành 128 bits data (không còn ở dạng ma trận)

Kết thúc quá trình mã hóa tại 1 round. Dữ liệu đầu ra của round này sẽ là dữ liệu đầu vào của round tiếp theo.

- Giải mã Decryption:

Quá trình giải mã sẽ là ngược lại của quá trình mã hóa. Số round cũng sẽ phụ thuộc vào độ dài khóa với 10, 12 hoặc 14 rounds

- Add round key
- Inverse MixColumns

Tương tự với MixColumns, nhưng sẽ khác ở ma trận thực thi

$$\begin{array}{c}
 [b_0] \\
 | b_1 | \\
 | b_2 | \\
 [b_3]
 \end{array}
 =
 \begin{array}{c}
 [14 \ 11 \ 13 \ 9] \\
 | 9 \ 14 \ 11 \ 13 | \\
 | 13 \ 9 \ 14 \ 11 | \\
 [11 \ 13 \ 9 \ 14]
 \end{array}
 \begin{array}{c}
 [c_0] \\
 | c_1 | \\
 | c_2 | \\
 [c_3]
 \end{array}$$

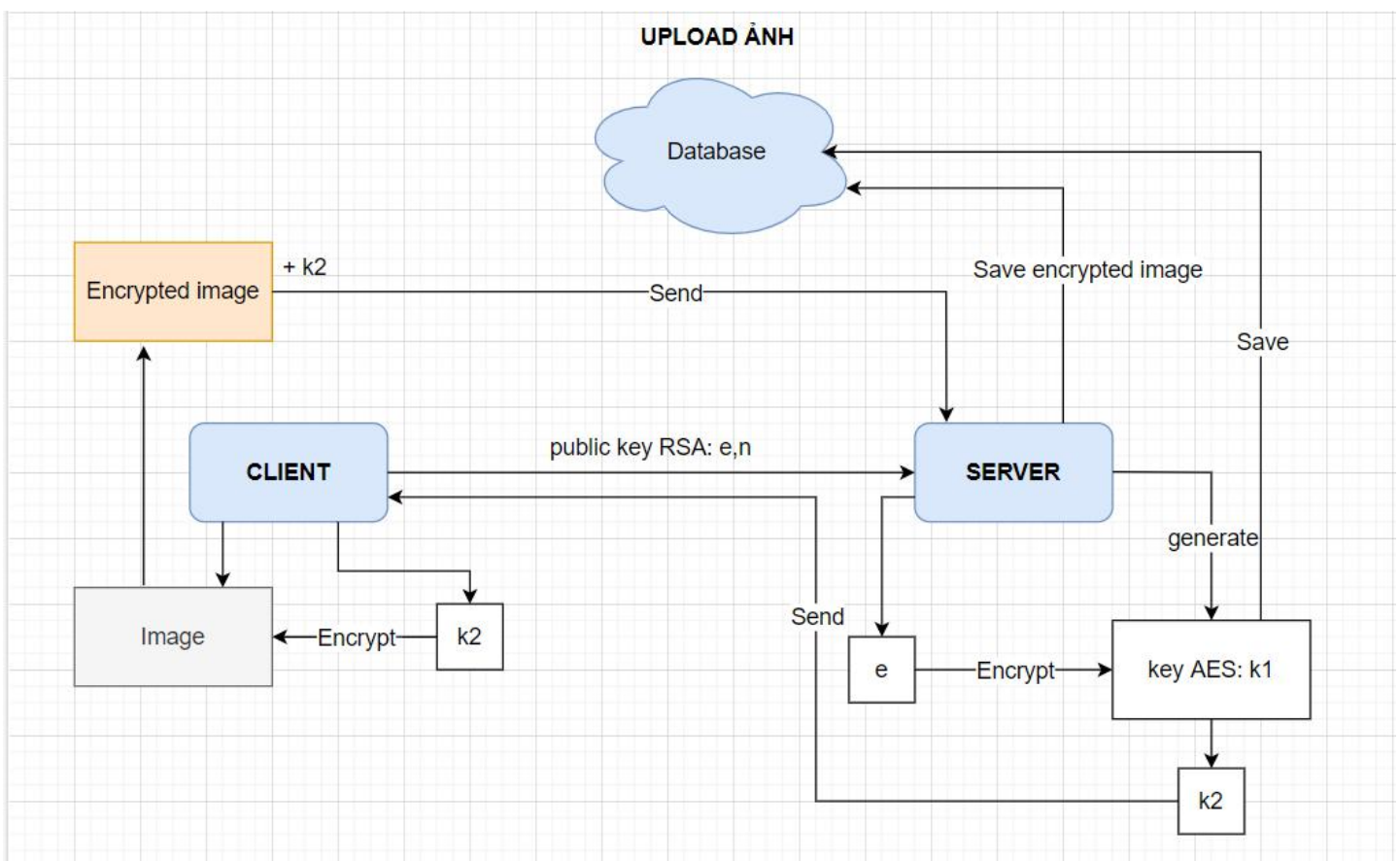
- ShiftRows
- Inverse SubBytes

Đảo ngược SubBytes bằng cách tra ngược lại bảng S-box đã sử dụng

2.2 Vận dụng

2.2.1. Mô tả tổng quan các dịch vụ:

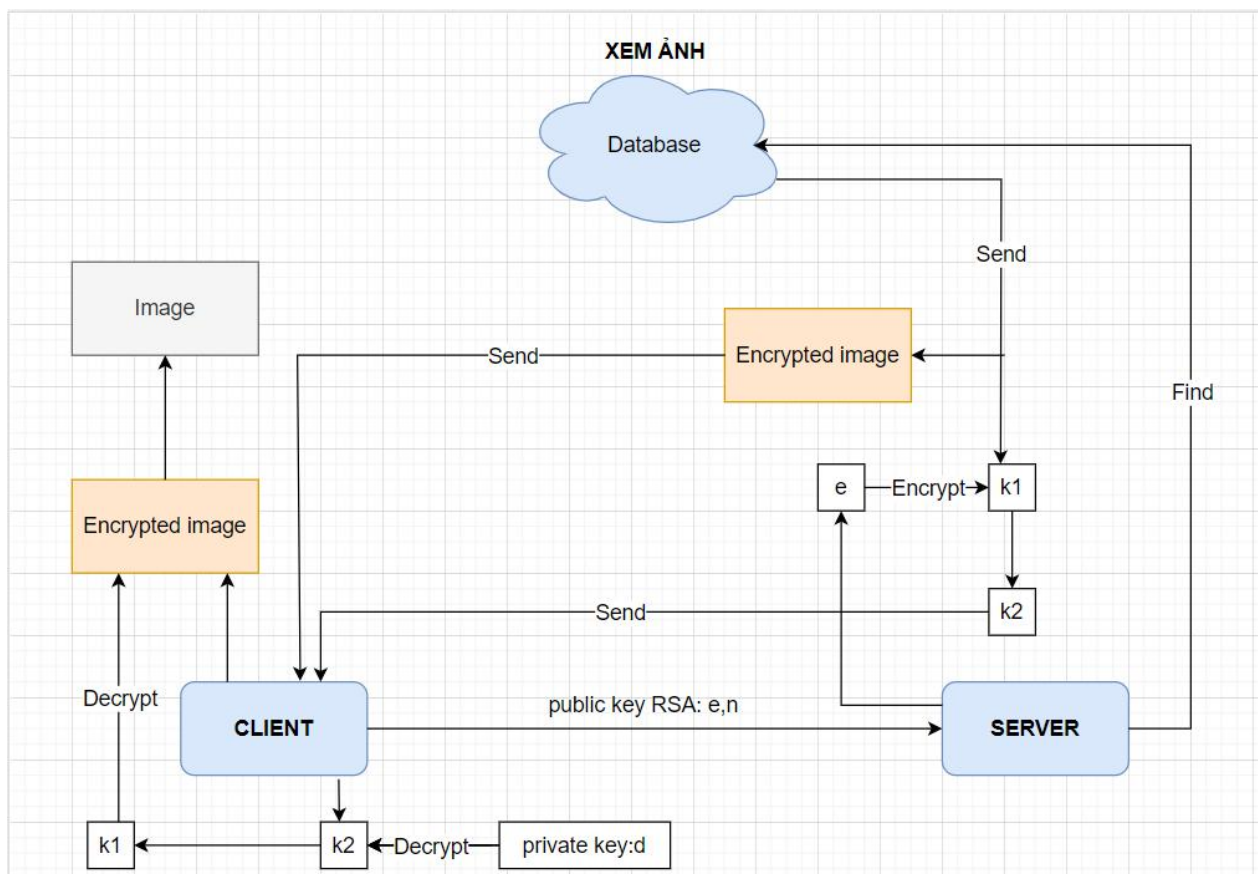
❖ Tải ảnh lên:



Mô tả:

- Client gửi public key RSA e cho Server để sử dụng
- Server generate 1 khóa AES $k1$ và lưu vào database. Sau đó dùng khóa e nhận được từ Client để mã hóa thành $k2$. Và gửi $k2$ cho phía Client.
- Client lấy key AES $k2$ nhận được từ Server, rồi dùng d và n giải mã thành $k1$ và lấy $k1$ để mã hóa cho ảnh
- Gửi ảnh đã mã hóa cùng với AES $k2$ lên cho server

❖ Xem ảnh:



Mô tả:

- Client gửi public key RSA e cho Server để sử dụng
- Server đọc từ database lấy ra key AES $k1$ và ảnh của người dùng muốn xem đã được mã hóa (được lưu từ quá trình upload ảnh trước đó). Dùng khóa (e và n) vừa nhận được để mã hóa $k1$ và gửi kèm cùng với ảnh đã được mã cho phía Client

- Client sau khi nhận được ảnh đã được mã và k2 sẽ dùng khóa bí mật RSA (d và n) để giải mã k2 thành k1.
- Sau đó, dùng khóa k1 này để giải mã ảnh và hiển thị cho người dùng.

❖ Chia sẻ ảnh:

Mô tả:

- Chủ sở hữu ảnh gửi yêu cầu chia sẻ lên server. Server sẽ thêm người dùng được thêm vào trong database.
- Thực hiện các bước tương tự như xem ảnh.

2.2.2. Các thành phần của source code:

- Folder server:

- + [app.py](#) là file server RestFul Api của ứng dụng
- + [rsa.py](#) có chức năng tạo khóa RSA, encrypt , decrypt văn bản
- + [requirements.txt](#) là file chứa các thư viện mà ứng dụng phụ thuộc

- Folder client

- + [assets](#) : là folder chứa các tài nguyên mà web sử dụng
- + [assets/js/RSA.js](#) : có chức năng tạo khóa RSA, encrypt , decrypt văn bản
- + [.html](#) : các file giao diện

2.2.3. Cách biên dịch và chạy chương trình:

- Khởi động server

- + Tại folder server, install đầy đủ các packet trong file [requirements.txt](#)
(pip install -r requirements.txt)
- + Tại folder server khởi chạy file [app.py](#)

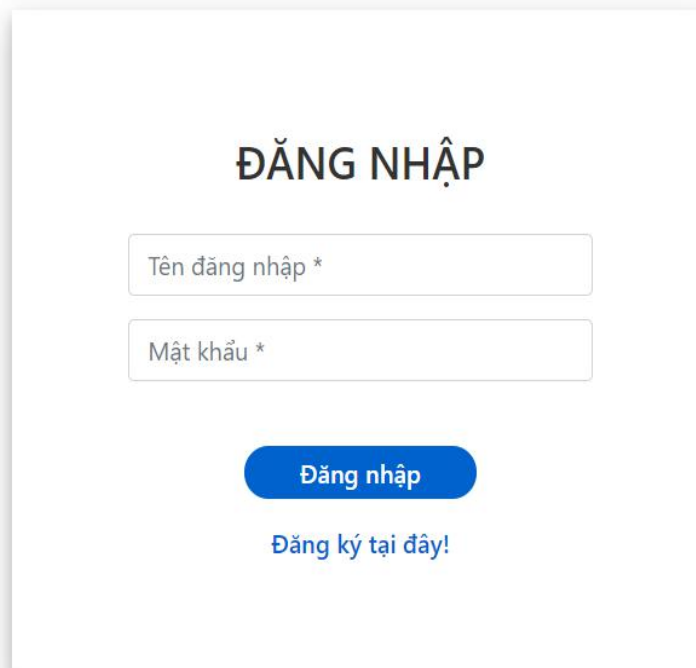
- Khởi động client

- + Tại folder client, mở 1 file .html bất kì bằng trình duyệt hoặc mở bằng Live server (extension của VScode)

2.2.4. Mã giả các thuật toán: (RSA, AES)

2.2.5. Hướng dẫn sử dụng chương trình:

- Sau khi chạy chương trình thành công, trình duyệt sẽ dẫn bạn đến trong đăng nhập (login). Tại đây bạn cần đăng nhập tài khoản của mình để sử dụng các dịch vụ của web. Nếu chưa có bạn nhấn vào “Đăng ký tại đây” để tạo một tài khoản của riêng mình.

A login form with a white background and a subtle shadow. At the top, the text "ĐĂNG NHẬP" is centered in a bold, black, sans-serif font. Below it are two input fields: the first is labeled "Tên đăng nhập *" and the second is labeled "Mật khẩu *". Both labels are in a light gray font. Below the input fields is a blue button with the text "Đăng nhập" in white. At the bottom, there is a link that says "Đăng ký tại đây!" in a blue font.

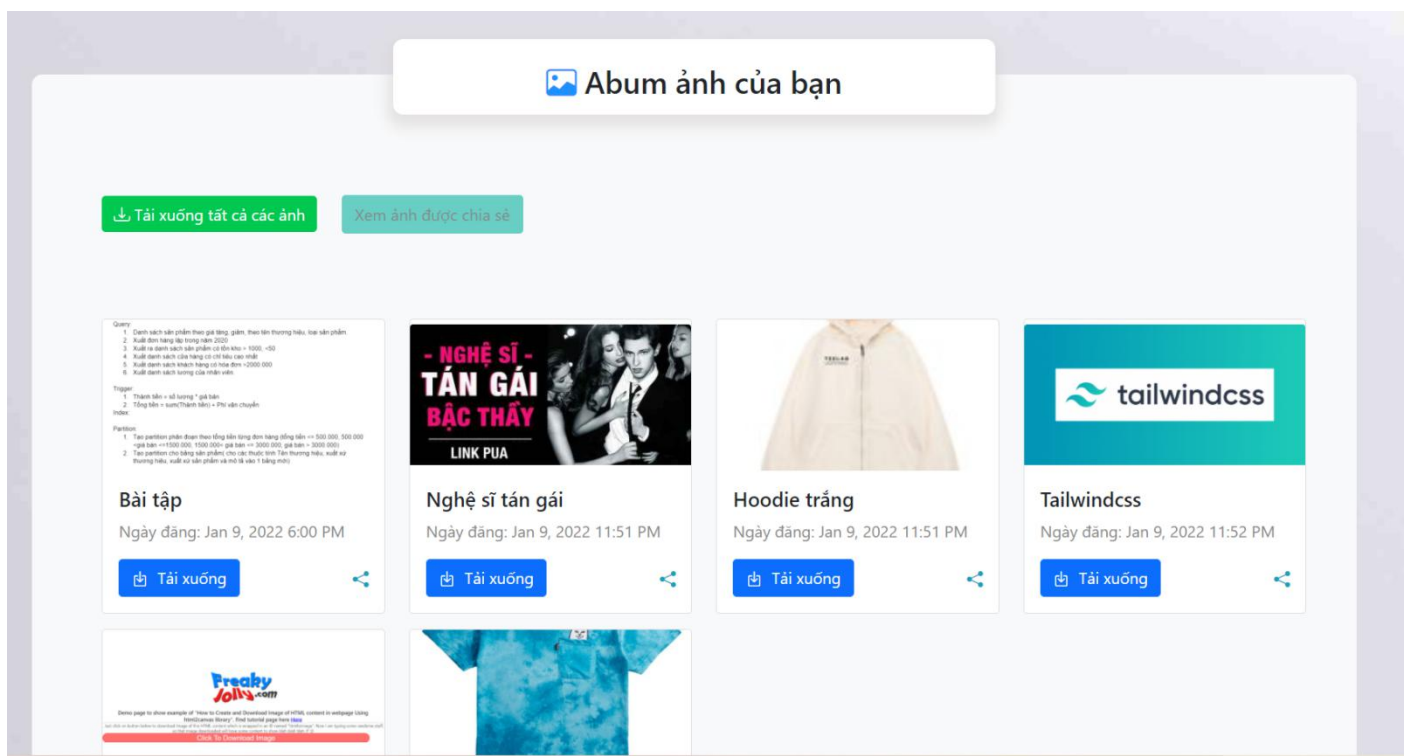
- Tại trang đăng ký, bạn điền đầy đủ các thông tin cần thiết theo mô tả và xác nhận. Nếu đăng ký thất bại trình duyệt sẽ hiện các thông báo lỗi và cần phải chỉnh sửa theo. Ngược lại trình duyệt sẽ chuyển hướng đến trang đăng nhập của website.

ĐĂNG KÝ

Đăng ký!

Bạn đã có tài khoản? [Đăng nhập ngay](#)

- Sau khi đăng nhập thành công, bạn sẽ đi đến giao diện chính của trang web. Tại đây bạn có thể thực hiện các dịch vụ được cung cấp như:



Xem album ảnh của bản thân

Tải ảnh lên



Preview ảnh



Đặt tên ảnh _____

LƯU ẢNH

Upload hình ảnh để lưu trữ tại trang web

Chia sẻ ảnh

Username

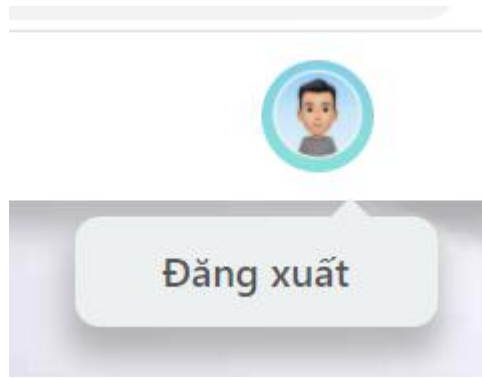
Username phải tồn tại .

Đóng

Gửi

Chia sẻ hình ảnh của mình với những người dùng khác

- Để kết thúc công việc hay thay đổi tài khoản khác bạn click vào biểu tượng user nằm trên góc phải màn hình và chọn “Đăng xuất”



2.3 Mở rộng

2.3.1. Ưu và nhược điểm của hệ thống:

❖ Ưu điểm:

- Có khả năng mã hóa được nhiều loại hình ảnh khác nhau như .png, .jpg,
- Dù hacker có thể bắt được tất cả gói tin cũng không xem được ảnh

❖ Nhược điểm:

- Khả năng đáp ứng truy cập đồng thời của nhiều người dùng một lúc.
- Thời gian mã hóa tốn nhiều thời gian

...

2.3.2. Công nghệ thế giới đang sử dụng

Về AES encryption:

Thuật toán AES (Advanced Encryption Standard) đã trở thành thuật toán mã hóa đặc lực cho nhiều khía cạnh quan trọng trong cuộc sống như các tổ chức tài chính, bảo mật doanh nghiệp,.. Thậm chí, Cơ quan An ninh Hoa Kỳ còn sử dụng thuật toán này để bảo vệ những thông tin “bảo mật quốc gia”

AES yêu cầu ít về các phép tính mã hóa. Nhờ vậy, nó mang lại tốc độ nhanh chóng, được ứng dụng trong nhiều thiết bị xách tay cũng như là trong việc mã hóa lượng dữ liệu lớn.

AES sử dụng hệ mã đối xứng, chỉ dùng 1 key cho việc mã hóa lẫn giải mã. Nó được cho là an toàn trước hầu hết các cuộc tấn công của những đối tượng xấu, ngoại trừ việc bị brute force, do nó có thể thử từng mã trong 128, 192 hoặc 256-bit cipher. Dự đoán trong tương lai, thuật toán AES sẽ còn được ứng dụng nhiều hơn nữa trong lĩnh vực bảo mật.

Về RSA encryption:

RSA là một hệ thuật toán mã hóa bất đối xứng, sử dụng một khóa công khai để mã hóa dữ liệu, và yêu cầu một khóa khác, được gọi là khóa bí mật, để có thể giải mã. Tuy công khai là vậy, nhưng lại không có cách để tính được khóa bí mật từ khóa công khai đã biết.

RSA phụ thuộc khá nhiều vào các phép toán chuyển đổi, do đó, nó sẽ có tốc độ khá chậm, thậm chí là không được sử dụng trong việc mã hóa một lượng dữ liệu lớn.

Một số thuật toán mã hóa khác:

DES: Một thuật toán mã hóa được giới thiệu từ những năm 1970 và được sử dụng rộng rãi trên phạm vi thế giới vào thời điểm đó. Tuy nhiên, đến hiện nay thì thuật toán này được đánh giá là thiếu tính bảo mật và không còn được ứng dụng rộng trong các lĩnh vực đời sống.

Triple DES: được thiết kế để thay thế cho thuật toán tiền thân của nó, DES. Bản thân nó cũng đã được sử dụng phổ biến trong công nghiệp nhưng cũng đang dần bị thay đổi bởi thuật toán tốt hơn là AES

Blowfish: cũng được thiết kế để thay thế cho thuật toán DES, blowfish nổi tiếng với tốc độ đáng kinh ngạc và sự hiệu quả rõ rệt mà nó mang lại. Blowfish còn được ứng dụng trong nhiều khía cạnh bảo mật như trên các nền tảng thương mại điện tử, nó có thể giúp bảo mật các thông tin thanh toán cũng như là mật khẩu người dùng. Đây là một trong những thuật toán vô cùng linh hoạt đã và đang được ứng dụng.

Với hướng đi hiện tại trong đồ án, là sử dụng RSA kết hợp với AES, tạo nên một hybrid cryptosystem (tạm dịch Hệ thống mật mã kết hợp), ta còn có thể mở rộng theo một số hướng khác như DES, Chaos-Maps,...

Với RSA + AES, hệ thống tận dụng được sự thuận tiện của hệ thống public key khi mà nó không yêu cầu người gửi lẫn người nhận phải chia sẻ cùng một mã khóa bí mật nhằm đảm bảo được sự bảo mật trong liên lạc. Tuy nhiên nó lại đòi hỏi nhiều các phép tính toán với độ phức tạp cao mà thậm chí còn không mấy là hiệu quả khi so sánh với hệ mã đối xứng (symmetric-key cryptosystems). Đó là lý do mà AES, một hệ mã đối xứng, được sử dụng nhằm giảm bớt sự bất lợi của hệ mã RSA.

2.3.3. Hướng cải tiến

Tại đây, chúng ta sẽ giới thiệu thêm một thuật toán: Genetic Algorithm nhằm kết hợp với 2 phương pháp RSA + AES để mang lại hiệu quả tốt hơn.

Thuật toán Genetic là một kỹ thuật được sử dụng trong khoa học máy tính để tìm ra giải pháp tối ưu nhằm thu gọn vấn đề. Thuật toán lần đầu được giới thiệu vào năm 1975, bởi John Holland và các cộng sự.

Áp dụng trong mã hóa và giải mã

Encryption: Tính chiều cao (H) và chiều rộng (W) của bức ảnh. Tìm $(H \bmod 8)$ và $(W \bmod 8)$, nếu bằng 0 thì sang bước tiếp.

$$\begin{aligned} H &= H + (8 - (H \bmod 8)) \\ W &= W + (8 - (W \bmod 8)) \end{aligned}$$

Chia bức ảnh làm 2 khối, mỗi khối có kích thước 8x8, sau đó thực hiện các phép biến đổi để nhận được một khối đã mã hóa. Lặp lại cho đến khi nhận được bức ảnh mã hóa.

Decryption: Lại đi vào từng khối của bức ảnh đang bị mã hóa, thực hiện các phép toán biến đổi để nhận lại khối ban đầu. Lặp lại cho đến khi nhận được bức ảnh đã giải mã.

Sơ đồ mã hóa:

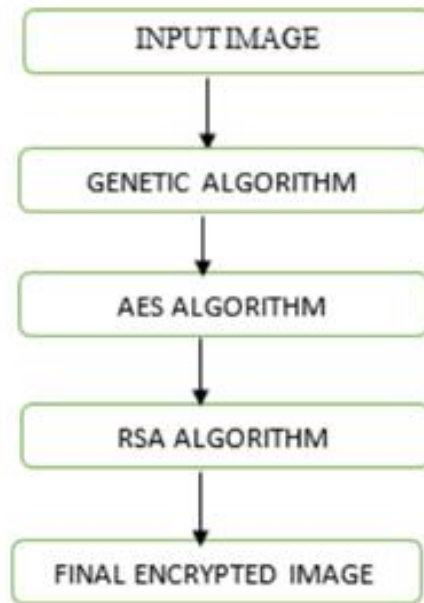


Figure 3. Flowchart for Hybrid Image Encryption

Sơ đồ giải mã:

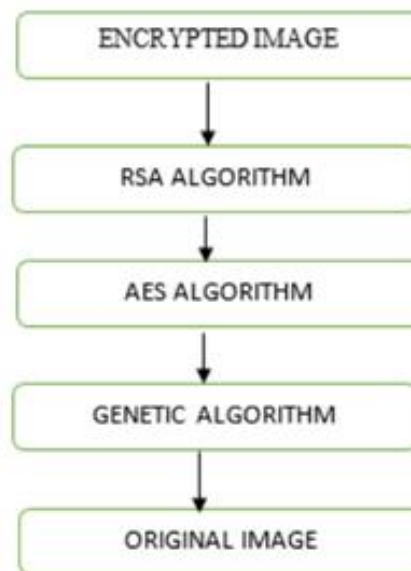


Figure 4. Flowchart for Hybrid Image Decryption

Ví dụ:



Figure 5. Test Image Lena Encryption. a) Original Image b) Genetic encryption c) AES encryption h) RSA encryption final image.

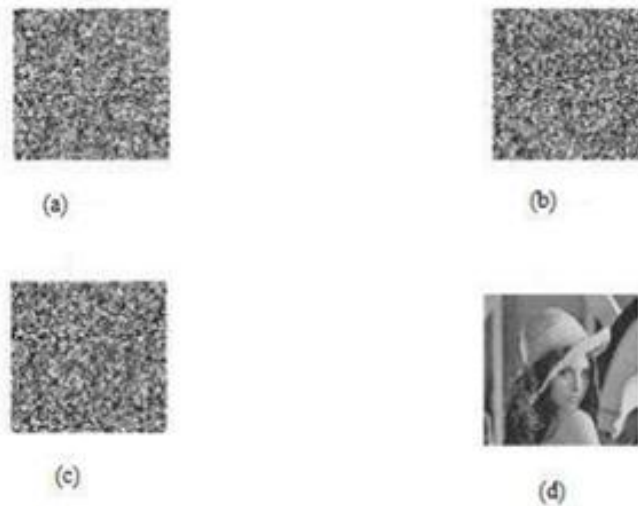


Figure 6. Test image Lena Decryption. a) RSA decryption b) AES decryption c) Genetic decryption d) original image.

3. Demo + Source



<https://www.youtube.com/watch?v=RGFDsQ9DK5k>



https://drive.google.com/drive/u/0/folders/11KdrfSKS1UpA3_g2cef5EldX8c7-osu



<https://github.com/ndthang000/RSA-python.git>

4. Tài liệu tham khảo

- A. Uk, Ijeacs. (2017). An Advance Approach of Image Encryption using AES, Genetic Algorithm and RSA Algorithm. International Journal of Engineering and Applied Computer Science (IJEACS). 02. 245-249. 10.24032/ijeacs/0208/01.
- B. [Advanced Encryption Standard \(AES\) - GeeksforGeeks](#)
- C. Picture of S-box in AES Algorithms explanation, [AES – Advanced Encryption Standard](#)